Character Error Correction for Chinese Speech Recognition System

SHEN Liqin CHAI Haixin** QIN Yong TANG Donald
IBM China Research Lab, 4/F, No.26, 6th Street, ShangDi, 100085, Beijing
** Computer Science Department, Tsinghua University
E-mail:shenlq@cn.ibm.com

ABSTRACT

With continuous speech recognition technology, one can input documents to computer very fast. Unfortunately, speech recognition can not achieve 100% accuracy, and users have to correct the recognition errors. Therefore the average throughput, which is the number of correct characters inputted to computer per minute taking the time for error correction into consideration also, drops dramatically. The smaller error rate and the better error correction mechanism, the less time will be required for error correction. Here we will focus on the second factor, the better error correction mechanism, to improve the throughput.

1. INTRODUCTION

In a word based continuous speech recognition system, one can keep a word-based candidate list for words in a decoding path. When correcting errors, if the correct word is in the candidate list, one can just say it or use the mouse to pick it. Therefore a good candidate list is very important for efficient error correction. However, the concept of a word is not well-defined. A word can be one or more characters. Word segmentation does not always lead to a unique word sequence. Given that there are many Chinese characters of the same or similar pronunciations, this implies potentially high word boundary errors in speech recognition results. We noticed in the decoding results from our system that, around 2/3 of decoding errors are word boundary ones, which means for these cases, the correct results can not be in the word candidate list from the engine. However, each character of the errors should have the similar pronunciation as the correct character even the whole word is wrong or the word boundary is wrong. In this paper, we present a method based on the notion of a "confusion matrix", to generate the character-based candidate list.

2. GENERATION OF THE CANDIDATES FOR EACH ERRONEOUSLY RECOGNIZED CHARACTER

We have a syllable set in Chinese defined as:

$SSet = \{S_1, S_2, \cdots, S_N\}$

In order to get the candidates for each decoded error E, we need to get the probability of each candidate character given the syllables of the decoded characters and its contexts, i.e.

$P(C \mid S_{HE}, H)$

Where C means a certain candidate, S_{HE} is the sequence of the syllables of the decoded characters, including the syllable of the decoded error itself and its nearest history, i.e.

$S_{HE} = S(H) + S(E)$

Where S(H) means the sequence of the syllables of H and S(E) means the syllable of E. H is its history in the language contexts. We then rank the candidates according to the value of above probability.

With Bayes's rule, we get

$$P(C \mid S_{HE}, H) = \frac{P(S_{HE}, H \mid C)P(C)}{P(S_{HE}, H)}$$

Because S_{HE} is a pure acoustic and H is a pure language event, we may regard them as independent variables. Further more, S_{HE} and H are determined for a given decoded character. So the above equation maybe simplified as

$$Rank \frac{P(S_{HE}, H \mid C)P(C)}{P(S_{HE}, H)} = Rank \frac{P(C \mid S_{HE})P(C \mid H)}{P(C)}$$

2.1. $P(C \mid S_{HE})$ —Generation and use of Confusion Matrix

For practical reasons, we simplify $P(C \mid S_{HE})$ as $P(C_S \mid S_E)$, where C_S represents the syllable of C, S_E is the syllable of the decoded error. The simplification indicates that we ignore the acoustic context $S(H)$ and group the characters with the same syllable into one class.

While training, we take M test speakers. Every speaker reads N test sentences. We decode these sentences for these speakers in syllables disregarding the language model.

For each syllable S_T in the test sentences, if it is decoded as S_D, where S_D could be S_T itself, we increment $Count(S_T \sim S_D)$ one step in the confusion matrix below. Then we can get the probability of S_T being recognized as S_D:

$$P(S_D \mid S_T) = \frac{Count(S_T \sim S_D)}{\sum_{S_m \in SSet} Count(S_T \sim S_m)}$$

Where $S_T, S_D \in SSet$, $Count(S_T \sim S_D)$ is the times of S_T being recognized as S_D.

$\sum_{S_m \in SSet} Count(S_T \sim S_m)$ is the summation of the row S_T, which means the total times S_T being recognized as any syllable $S_m \in SSet$. We can save $P(S_D \mid S_T)$ in the final confusion matrix.

At the same time, we can get:

$$P(S_T) = \frac{Count(S_T)}{\sum_{S_m \in TrainingData} Count(Sm)} \qquad (*)$$

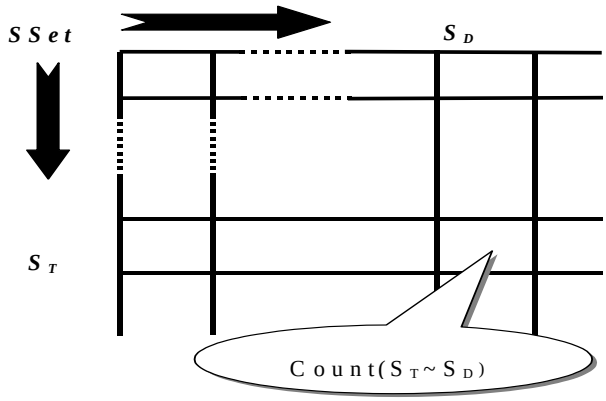

Figure 1. Confusion Matrix

While using the confusion matrix, we are given the decoded syllable S_D, and we want to get the probability S_D from a certain original test S_T, i.e.$P(S_T|S_D)$. With Bayes's rule:

$$P(S_T \mid S_D) = \frac{P(S_D \mid S_T)P(S_T)}{P(S_D)}$$

When we calculate $P(C_S \mid S_E)$,

$$P(C_S \mid S_E) = P(S_T = C_S \mid S_D = S_E)$$

$$= \frac{P(S_D = S_E \mid S_T = C_S)P(S_T = C_S)}{P(S_D = S_E)}$$

$P(S_D = S_E)$ is the same for all candidates, so it is no need for ranking. We can get $P(S_D = S_E \mid S_T = C_S)$ from the confusion matrix and $P(S_T = C_S)$ in equation (*).

2.2. P(C|H)/P(C)—Language Model(LM) to restrict the candidate list

To restrict the candidate list within a reasonable length while keeping the correct choice in, it's very natural to use LM to prune the list, i.e. to include the term P(C|H)/P(C) for better ranking.

2.2.1. Character LM

To apply character LM to get the P(C|H). We can use trigram:

$$P(C \mid H) \approx P(C \mid C_{-2}, C_{-1})$$

Where C_{-2}, C_{-1} are the previous 2 characters of C.

2.2.2. Word LM

Because we have word history of the current error, we can take advantage of word LM expecting to get better prediction of the current correct choice.

$$P(C \mid H) \approx P(C \mid W_{-2}, W_{-1})$$

Where W_{-2}, W_{-1} are the previous 2 words of C. However, when we build word LM, there is no probability of $P(C \mid W_{-2}, W_{-1})$, so we have to estimate:

$$P(C \mid W_{-2}, W_{-1}) = \sum_{allW_C} P(W_C \mid W_{-2}, W_{-1})$$

Where W_C is one of the words whose first characters are C.

If C is the second character of a error word, and W_C=C_{-1}C is a real word, then:

$$P(C \mid W_{-2}, W_{-1}, C_{-1}) = \sum_{allW_{WC}} P(W_{W_C} \mid W_{-2}, W_{-1})$$

Where W_{W_C} is one of the words whose first 2 characters are C_{-1}C

If C is the second character of a error word, and W_C=C_{-1}C is not a legal word, then:

$$P(C \mid W_{-2}, W_{-1}, C_{-1}) = \sum_{allW_C} P(W_C \mid W_{-2}, W_{-1})$$

Where W_C is one of the words whose first characters are C

For the 3rd or later characters in errors, we can deduce the similar formula also.

2.3. Weighting between $P(C_S \mid S_E)$ and $P(C \mid H)/P(C)$

	LM Base Unit	Toned/ Untoned	Weight of Acoustic	Top 10	Top20	Top50	Top 100
1	Character	Untoned	0.5	59.12%	67.91%	78.14%	83.77%
2	Character	Toned	0.5	56.36%	65.05%	73.59%	77.83%
3	Character	Untoned	0	1.57%	2.49%	5.63%	11.73%
4	Character	Untoned	1	1.63%	2.50%	5.96%	12.23%
5	Character	Untoned	0.2	59.12%	67.91%	78.14%	83.77%
6	Character	Untoned	0.8	59.12%	67.91%	78.14%	83.77%
7	Character	Untoned	1.00E-03	59.12%	67.91%	78.14%	83.77%
8	Character	Untoned	1.00E-10	59.12%	67.91%	78.14%	83.77%
9	Word	Untoned	0.5	49.79%	60.41%	72.75%	80.84%
10	Word	Toned	0.5	48.88%	58.65%	70.26%	76.27%
11	Word	Toned	0.1	49.07%	58.93%	70.48%	76.64%
12	Word	UnToned	0.1	49.89%	60.34%	72.91%	81.08%

Table 1. Comparison Results

We maybe weight $P(C_S \mid S_E)$ and $PC \mid H)/P(C)$ differently to balance acoustic and language information:

$Rank[P(C \mid S)]^{\alpha}[P(C \mid H)/P(C)]^{(1-\alpha)}$

Where $0 < \alpha < 1$, we can tune α to achieve the best candidate list.

3. EXPERIMENT RESULTS

We took 200 speakers, 50 each from Beijing, Shanghai, GuangZhou and SiChuan regions. Each speaker read 100 sentences. We decoded the sentences and use the decoding results as our training data to build the confusion matrix.

We took 20 test speakers, each speaker read 100 sentences respectively. For the decoding errors, we generate the candidate list for each error with the method proposed above, and more than 60% of the total errors, we can find the correct character in the top 10 list.

We did more experiment and analysis on Toned and Toneless Confusion Matrix**,** Character and word LM**,** different size of LM, and played with α .

4. CONCLUSION

For Chinese, character level error correction is useful to complement the word level error correction method due to the fact that decoding word boundaries are often wrong. . The generation of character candidate list proposed here shows more than of 60% of decoding errors, we can get the correct character in the top 10 candidate list, while less than 30% of errors, we can get the correct word in the word candidate list. This makes the error correction mechanism for Chinese much more efficient.

From Table 1, we can draw the following conclusion:

- **Conclusion 1:** ***Untoned confusion matrix has more information for error correction***. Refer to result 1 vs. result 2, result 9 vs. result 10, result 11 vs. result 12
- **Conclusion 2:** ***The actual weight value is not important, but can not be ignored***. Refer to result 1, 3, 4, 5, 6, 7 and 8.
- **Conclusion 3:** ***Character based LM has better performance,*** though it seems word based LM should be better. Maybe the reason is current word based LM is not sufficient. Refer to result 1,2 and result 9,10.